filter 202 reserves a portion of the name space of documents which can be requested using a URL for RPC requests. As described more completely below, URL filter 202 determines whether a particular URL specifies a document in the reserved name space portion and processes the URL accordingly. In accordance with HTTP, applet 200 sends a URL specifying a document to RPC process 210 and receives the specified document from RPC process 210. To invoke either of RPC functions 206A-B, applet 200 forms a URL according to the steps of logic flow diagram 300 (Figure 3) and sends the URL to RPC process 210.

Substitute the paragraph starting at page 12, line 17, with the following:

Applet 200 (Figure 2) can make itself available to receive RPC requests from RPC process 210 in a manner which is generally permitted by applet viewer 150 (Figure 1) and which is illustrated in logic flow diagram 700 (Figure 7). Processing according to logic flow diagram 700 begins in step 702.

Substitute the paragraph starting at page 12, line 21, with the following:

In step 702, applet 200 (Figure 2) builds an RPC request for execution of an "RPC ready" RPC function by RPC process 210 and encodes the RPC request as a URL in the manner described more completely above. In step 704 (Figure 7), applet 200 (Figure 2) sends the URL encoded in step 702 (Figure 7) to RPC process 210 to thereby request execution of the "RPC ready" RPC function, which can be RPC function 206B, for example.

<u>Substitute the paragraph starting at page 12, line 26, with the following</u>:

The design and implementation of RPC function 206B is such that execution thereof indicates to RPC process 210 that applet 200 is ready to receive RPC requests from RPC process 210 and establishes a communications channel through which RPC process 210 can send RPC requests to applet 200. Specifically, HTTP, as implemented by both RPC process 210 and applet viewer 150 (Figure 1) within which applet 200 executes, expects a document to be retrieved in response to the URL sent in step 704 (Figure 7 4). In addition, HTTP as implemented permits transfer of the requested document to be delayed and intermittent. However, RPC process 210 requests a virtual document, i.e., a document which does not exist within memory 104 (Figure 1) of computer system 100 but which is instead created in response to the URL. Execution of RPC function 206B (Figure 2) of RPC process 210 changes the state of RPC process 210 to indicate that applet 200 is ready to receive RPC requests and to store data identifying the communications channel through which applet 200 is waiting to receive a document in response to the URL sent in step 704 (Figure 7).

<u>Substitute the paragraph starting at page 13, line 10, with the following</u>:

RPC process 210 includes a core function 208 which defines and implements a central task for which RPC process 210 is designed. Execution of core function 208 can include sub-tasks which are implemented by one or more of RPC functions 212 of applet 200. Accordingly, to cause performance of such sub-tasks, core function 208 of RPC process 210 builds RPC requests which request execution of a selected one of RPC functions 212 and includes zero or more parameters to be used by the selected RPC function as input data. To send

such an RPC request to applet 200, RPC process 210 sends the RPC request to applet 200 as a portion of the virtual document requested by the URL sent by applet 200 in step 704 (Figure 7). By sending the RPC request as only a portion of the requested virtual document, RPC process 210 (Figure 2) indicates to applet 200 that other RPC requests can be subsequently sent to applet 200 through the same communication channel. Since the RPC request is sent to applet 200 as part of a document, the contents of which are not constrained by any particular protocol such as HTTP, the RPC request can be in any convenient form and can be in a form which is entirely inappropriate for a HTTP URL. To terminate the communication channel, and therefore terminate the ability of applet 200 to receive RPC requests from RPC process 210, RPC process 210 sends data indicating that the entirety of the virtual document requested by applet 200 has been sent to applet 200.

Substitute the paragraph starting at page 13, line 26, with the following:

Processing by applet 200 transfers from step 704 (Figure 7) to loop step 706 in which steps 708-712 are performed repeatedly until applet 200 (Figure 2) receives data indicating that the entirety of the requested virtual document has been received. In step 708 (Figure 7), applet 200 (Figure 2) receives a portion of the virtual document from RPC process 210. In step 710 (Figure 7), applet 200 (Figure 2) parses an RPC request from the received portion. As described above, the format of the RPC request can be entirely independent of the format of HTTP URLs. In step 712 (Figure 7), applet 200 (Figure 2) services the parsed RPC request by executing one of RPC functions 212 specified by the parsed RPC request and supplying any arguments parsed from the received portion as input

data. Any results produced by servicing the parsed RPC request can be communicated to RPC process 210 in the form of an HTTP URL built and sent to RPC process 210 in the manner described above. The results URL identifies the RPC functions 212 invoked by the parse RPC request to specify to RPC process 210 to which RPC request the resulting data pertains.

Substitute the paragraph starting at page 14, line 10, with the following:

Steps 708-712 (Figure 7) are repeated until applet 200 (Figure 2) receives data from RPC process 210 indicating that the entirety of the requested virtual document has been sent to applet 200. Thereafter, processing according to logic flow diagram 700 completes.

Substitute the paragraph starting at page 14, line 13, with the following:

In this way, applet 200 accepts RPC requests from RPC process 210 in a manner which is permitted by applet viewer 150 (Figure 1) without requiring modification of applet viewer 150. Accordingly, many of the advantages of interprocess communication are achieved in the secure context of an applet viewer.